Revised reprint (Dec. 1997) of *Graphical Models and Image Processing* **59** (1997), 89–96.
/engc/press/toffoli-rot/rot-gmip

Three-dimensional rotations by three shears

Tommaso Toffoli
tt@bu.edu, Electrical and Computer Engineering, Boston University
8 Saint Mary's St., Boston, MA 02215.
Jason Quick
quick@gte.net, 8222 Regents Road #204, San Diego, CA 92122

Received December 15, 1995; revised September 6, 1996; accepted January 17, 1997

We show that a rotation in three dimensions can be achieved by a composition of three shears, the first and third along a specified axis and the second along another given axis orthogonal to the first; this process is invertible. The resulting rotation algorithm is practical for the processing of fine-grained digital images, and is well adapted to the access constraints of common storage media such as dynamic RAM or magnetic disk.

For a 2-D image, rotation by composition of three shears is well known. For 3-D, an obvious nine-shear decomposition has been mentioned in the literature. Our three-shear decomposition is a sizable improvement over that, and is the best that can be attained—just two shears won't do.

Also, we give a brief summary of how the present three-shear decomposition approach generalizes to any linear transformations of unit determinant in any number of dimensions.

1 Introduction

In two dimensions, a rotation can be expressed as a composition of three shears along two given orthogonal axes (see Paeth[5] and Tanaka et. al.[8]), as illustrated below.

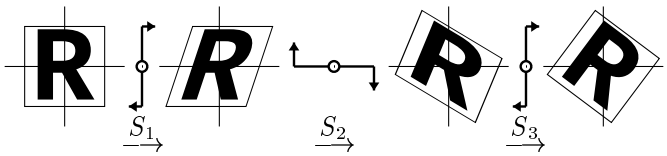

Figure 1: Two-dimensional rotation achieved by a succession of three shears.

In three dimensions, rotations and shears form a much more complex picture; generalizations from two dimensions are not automatic.

Since a rotation in three-dimensions can be written as a composition of three rotations about orthogonal axes, our starting point for a three-dimensional rotation was a composition of *nine* shears (cf. [11]). Our main result is that one can express a rotation in three dimensions as the composition of just *three* shears. As in the two-dimensional case, each of these shears can be viewed as a parallel transport of "pixel rows" that are oriented parallel to one of the three orthogonal axes.

Algorithms that achieve spatial transformations by means of shears are useful in many computer graphics contexts, as discussed in [5, 9]. Briefly, when a discrete pixel array is subjected to a continuous address transformation the transformed addresses usually fall between array points; to construct the new pixel value at a given array point one must take a suitably weighted average of the pixels that fell near that point. Though shears are continuous transformations, they maintain uniform pixel spacing along each coordinate axis. Thus, it becomes possible to approximate a shear by a discrete, one-to-one address transformation whereby each new pixel falls within just one pixel spacing of the ideal position specified by the shear itself. In this case, if the image resolution is high enough, there is no need to "look inside the pixel" and do *value* arithmetic on their contents: it is enough to ship the pixels to their destinations in a "data-blind" fashion, doing just *address* arithmetic.

The first and third shear of our decomposition $R = S_3 S_2 S_1$ can be oriented so as not to disturb the natural ordering of pixels on the mass-storage medium (disk or tape track, dynamic RAM row); this is called *scanline order* in the literature[1]. Thus, for these shears most data movements reduce to sequential-access operations ("string copy with offset"), which can exploit the medium's full bandwidth.

The second shear will not have the advantage of sequential access, since it must run "across the grain" of memory (some such operation is unavoidable if the end-result is to be a rotation). The advantage remains, in all cases, that the address arithmetic involved in a shear is much simpler than that of a rotation; this makes it possible to handle a shear directly at the memory-controller level, where it can be done more efficiently than at the processor level.

Rotations through shears are particularly efficient in the CAM-8 architecture[4], where "matrix copy with index offset" (in any number of dimensions) is one of the native operations.

2 Rotations

A rotation in the plane by an angle ψ, denoted by $\mathrm{R}(\psi)$, is expressed as a matrix (to operate from the left on column vectors) as follows

$$\mathrm{R}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}.$$

In three dimensions, besides the rotation angle ψ one has to specify the axis $\hat{n}$ about which the rotation takes place.

The corresponding rotation is denoted by $\mathrm{R}_{\hat{n}}(\psi)$. In a given frame, $\hat{n}$ can be specified in terms of a *polar* angle θ and an *azimuthal* angle ϕ.

Rotations about the three coordinate axes are expressed as follows

$$\mathrm{R}_1(\psi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & -\sin\psi \\ 0 & \sin\psi & \cos\psi \end{bmatrix},$$

$$\mathrm{R}_2(\psi) = \begin{bmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{bmatrix},$$

$$\mathrm{R}_3(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The three angles θ, ϕ, ψ give a parametrization of the group of rotations, SO(3). Another parametrization, in terms of the three *Euler angles*, α, β, and γ, is defined by the following construction. Given the frame (1,2,3),

- Rotate this frame by an angle α about axis 3, yielding a new frame $(1\mathrm{a}, 2\mathrm{a}, 3)$.
- Rotate the new frame by an angle β about the 2a axis, yielding a new frame $(1\mathrm{b}, 2\mathrm{a}, 3')$.
- Rotate the new frame by an angle γ about $3'$, yielding the final frame $(1', 2', 3')$.

Thus, the overall rotation can be written as

$$\mathrm{R}(\alpha,\beta,\gamma) = \mathrm{R}_{3'}(\gamma)\mathrm{R}_{2\mathrm{a}}(\beta)\mathrm{R}_3(\alpha).$$

It turns out that $\mathrm{R}(\alpha,\beta,\gamma)$ can be rewritten in terms of three rotations about only *two* of the original axes (namely, 2 and 3), using the same angles α, β, and γ in the following order

$$\mathrm{R}(\alpha,\beta,\gamma) = \mathrm{R}_3(\alpha)\mathrm{R}_2(\beta)\mathrm{R}_3(\gamma) = Q_3Q_2Q_1. \qquad (1)$$

In matrix form, $\mathrm{R}(\alpha,\beta,\gamma)$ is expressed as follows (where 'c' and 's' stand for 'cos' and 'sin')

$$\mathrm{R}(\alpha,\beta,\gamma) = \begin{bmatrix} \mathrm{c}\alpha\mathrm{c}\beta\mathrm{c}\gamma - \mathrm{s}\alpha\mathrm{s}\gamma & -\mathrm{c}\alpha\mathrm{c}\beta\mathrm{s}\gamma - \mathrm{s}\alpha\mathrm{c}\gamma & \mathrm{c}\alpha\mathrm{s}\beta \\ \mathrm{s}\alpha\mathrm{c}\beta\mathrm{c}\gamma + \mathrm{c}\alpha\mathrm{s}\gamma & -\mathrm{s}\alpha\mathrm{c}\beta\mathrm{s}\gamma + \mathrm{c}\alpha\mathrm{c}\gamma & \mathrm{s}\alpha\mathrm{s}\beta \\ -\mathrm{s}\beta\mathrm{c}\gamma & \mathrm{s}\beta\mathrm{s}\gamma & \mathrm{c}\beta \end{bmatrix}. \qquad (2)$$

3 Rotations by shears

A rotation in the plane by an angle $\psi \neq \pi$ can be written as a composition of three shears, each along one of the two axes and across the other (cf. Fig. 1), namely,

$$\mathrm{R}(\psi) = S_3S_2S_1 = \begin{bmatrix} 1 & -\tan\frac{\psi}{2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \sin\psi & 1 \end{bmatrix} \begin{bmatrix} 1 & -\tan\frac{\psi}{2} \\ 0 & 1 \end{bmatrix}, \qquad (3)$$

One can apply decomposition (3) separately to the three rotations in (1), obtaining the shear products

$$\begin{aligned} Q_1 &= S_{13}S_{12}S_{11} \\ &= \begin{bmatrix} 1 & -\tan\frac{\gamma}{2} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ \sin\gamma & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -\tan\frac{\gamma}{2} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\ Q_2 &= S_{23}S_{22}S_{21} \\ &= \begin{bmatrix} 1 & 0 & \tan\frac{\beta}{2} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\sin\beta & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & \tan\frac{\beta}{2} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\ Q_3 &= S_{33}S_{32}S_{31} \\ &= \begin{bmatrix} 1 & -\tan\frac{\alpha}{2} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ \sin\alpha & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -\tan\alpha 2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \end{aligned} \qquad (4)$$

As a result, $Q = \mathrm{R}(\alpha,\beta,\gamma)$ can be written as the product of nine shears as follows

$$Q = Q_3Q_2Q_1 = S_{33}S_{32}S_{31}\ S_{23}S_{22}S_{21}\ S_{13}S_{12}S_{11}, \qquad (5)$$

as remarked by Voss[11]. Can one write Q as the product of fewer than nine shears?

Each of the nine matrices in (4) differs from the identity matrix by a single element, on row u and column v; this element is a parameter that specifies the rate of shearing *along* axis u and *across* axis v. For our purposes we shall need unrestricted shears, characterized by three parameters (see §6 for a fuller discussion). The most general shear along axis 1 is represented by a triangular matrix with unit diagonal,

$$S = \begin{bmatrix} 1 & a & b \\ 0 & 1 & c \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & a & b \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = S_{\mathrm{T}}S_{\mathrm{L}}.$$

This can be thought of as the composition of two transformations, S_{L}—which takes image lines parallel to axis 1 and shifts them along themselves—and S_{T}—which shifts those lines sideways in the direction of axis 2, while preserving the first and third coordinates.

Making use of such unrestricted shears, one can rearrange the factors of (5) aiming to bring to adjacent positions shears performed along the same axis. This task was undertaken by the second author (Quick). The resulting process, laborious but straightforward, is described in §3.1. Eventually, recombination of adjacent terms leads to the product of just three shears, namely,

$$Q = S_3S_2S_1, \qquad (6)$$

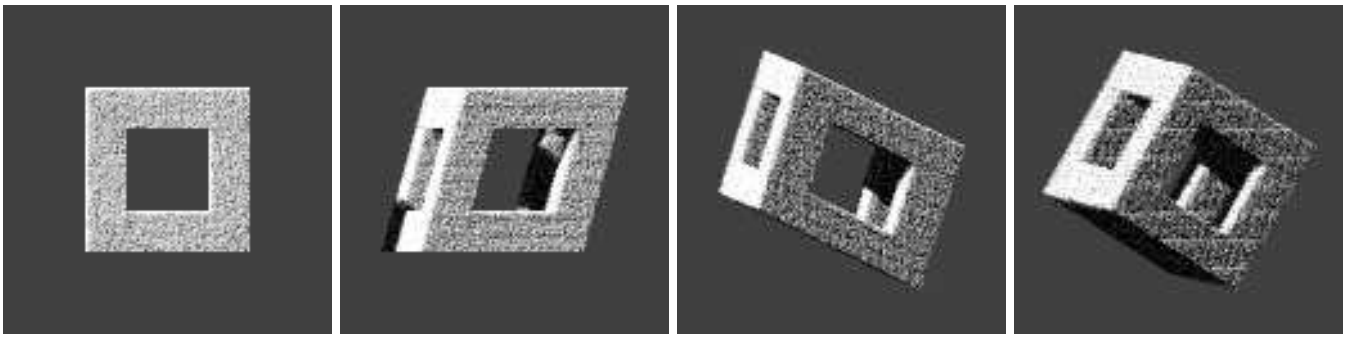

Figure 2: Three-dimensional rotation (Euler angles: $\alpha = -10^\circ$, $\beta = 30^\circ$, $\gamma = -15^\circ$) achieved by a succession of three shears. See Footnote 3 at the end of §5 for details.

with

$$S_1 = \begin{bmatrix} 1 & -\tan\frac{\alpha+\gamma}{2} & \frac{\cos\frac{\alpha-\gamma}{2}}{\cos\frac{\alpha+\gamma}{2}}\tan\frac{\beta}{2} \\ 0 & 1 & -\sin\gamma\tan\frac{\beta}{2} \\ 0 & 0 & 1 \end{bmatrix},$$

$$S_2 = \begin{bmatrix} 1 & 0 & 0 \\ \sin(\alpha+\gamma) & 1 & 0 \\ -\cos\gamma\sin\beta & -\frac{\sin\frac{\alpha-\gamma}{2}}{\cos\frac{\alpha+\gamma}{2}}\sin\beta & 1 \end{bmatrix}, \qquad (7)$$

$$S_3 = \begin{bmatrix} 1 & -\tan\frac{\alpha+\gamma}{2} & \cos\alpha\tan\frac{\beta}{2} \\ 0 & 1 & \sin\alpha\tan\frac{\beta}{2} \\ 0 & 0 & 1 \end{bmatrix}.$$

The successive action of these shears is illustrated in Fig. 2. Such a decomposition exists (except for the degenerate cases $\alpha+\gamma = \pi$ or $\beta = \pi$) for any given choice of orthogonal axes 1, 2 and 3 as a coordinate system.

3.1 Construction of S_1, S_2, S_3 from Q

Fig. 3 illustrates the process by which the three shear matrices S_1, S_2, S_3 are obtained from the rotation matrix Q. Proceeding from the top downwards, a branch point (denoted by a hollow dot) indicates that the incoming matrix is factored into the product of the three outgoing matrices; conversely, a merge point (denoted by a solid dot) indicates that the two incoming matrices are multiplied together. At a nondotted crossing, the idea is to "change the order of the two factors"; what actually happens is that the matrix on the vertical path goes through unchanged while that on the horizontal path moves over the first and in the process is *similarity transformed* by it (that is, in the product AB, factor B can "move over A", but at the cost of turning into $B' = ABA^{-1}$).

Specifically, the given rotation Q is factored as $Q_3Q_2Q_1$ (three 2-D rotations about axes 3, 2, and 3 respectively, as in (1)); Q_2 is further factored as $S_{23}S_{22}S_{21}$ (three 2-D shears as in (4)), while Q_3 and Q_1 are multiplied to yield Q_4, which is still a 2-D rotation about axis 3. The latter is further factored into three 2-D shears, using the generic formula (3). We now perform all of the path crossings: S_{21} is transformed by S_{21} into $U_1 = Q_1^{-1}S_{21}Q_1$; S_{23} into U_3; and S_{22}, after two crossings, into S_b. Finally, we recombine the resulting six pieces (U_3, S_{43}; S_{42}, S_b; S_{41}, U_1) two by two, yielding the desired three shears S_3, S_2 and S_1.

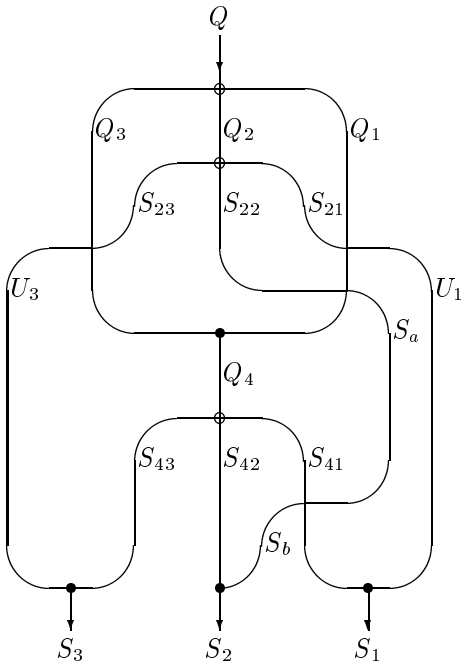

Figure 3: Construction of the shears S_1, S_2, S_3 of (6), starting from the rotations Q_1, Q_2, Q_3 of (4). A hollow dot indicates that a matrix is decomposed into the product of three matrices; a solid dot, that two matrices are multiplied together. At a nondotted crossing, the matrix on the vertical path goes through unchanged while that on the horizontal path moves over it and bears the attendant commutation burden; for example, when S_{21} crosses the path of Q_1 it becomes $U_1 = Q_1^{-1}S_{21}Q_1$.

3.2 Minimum number of shears

It is clear that *at least three* shears are needed to synthesize a rotation. In fact, the product of two shears has the form

$$P = \begin{bmatrix} 1 & 0 & 0 \\ a_{21} & 1 & 0 \\ a_{31} & a_{32} & 1 \end{bmatrix} \begin{bmatrix} 1 & b_{12} & b_{13} \\ 0 & 1 & b_{23} \\ 0 & 0 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} 1 & b_{12} & b_{13} \\ a_{21} & a_{21}b_{12}+1 & a_{21}b_{13}+b_{23} \\ a_{31} & a_{31}b_{12}+a_{32} & a_{31}b_{13}+a_{32}b_{23}+1 \end{bmatrix}.$$

If P is to be a rotation, normality of the first row and column demands that

$$a_{21} = a_{31} \equiv 0,$$
$$b_{12} = b_{13} \equiv 0.$$

Thus,

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & b_{23} \\ 0 & a_{32} & 1+a_{32}b_{23} \end{bmatrix};$$

by similarly requesting normality for the second row and column, one concludes that $P \equiv I$ (this iterative argument generalizes to any number of dimensions). Thus, the *only* rotation that can be achieved by the product of two shears is the identity.

If one allows transformations that *stretch* an image at the same time as they shear it, then a 2-D rotation can be achieved by the product of just *two* such transformations[1, 6]. The transformations used for this purpose are *pseudo-shears*, i.e., matrices that differ from the identity matrix in just one row. With three pseudo-shears one can synthesize a rotation in three dimensions[2]; in general, n pseudo-shears suffice for an n-dimensional rotation. Even though their determinant is not restricted to be unit, pseudo-shears are in certain respects simpler than the unrestricted shears employed in this paper (cf. §6.1), since they are described by n parameters rather than $n(n-1)/2$. With them, however, one loses the advantage of a one-to-one pixel correspondence under the transformation (cf. next section).

4 Rotation of digitized images; error containment

In a digitized image, both pixel addresses and contents are restricted to discrete sets of values. Thus, a rotation may entail a resampling of the image—a likely source of errors. Even though resampling and anti-aliasing are important concerns, a full discussion of these issues is not among the goals of the present paper; here we shall only briefly discuss certain ways to take advantage of the original features of the present approach to rotations.

When we apply a continuous geometric transformation T to the integer vector u that specifies the site of a pixel in an array, the resulting vector $v = Tu$ in general will not have integer components. Conversely, if we choose an array site as the destination address v, the source address $u = T^{-1}v$ will fall in between array sites. A typical solution to this problem is to interpolate the data, i.e., construct the new pixel at v as a suitably weighted average of the pixels near u. In fact, the original work on 2-D rotations by shears was motivated by the simplified filtering that would be required (basically involving only 1-D rather than 2-D Fourier analysis; see [12] for sample code). This issue is amply discussed in [5] and references therein.

As mentioned in §1, a cheaper approach is to do *address roundoff* rather than *data interpolation*. With transformations that do not conserve area, this entails decimation or repetition of pixels. However, since shears are nonscaling invertible tranformations (the have unit determinant), the address roundoff algorithm can be designed in such a way that, upon each of the three shears and for each of the two coordinates affected by a shear,

- each pixel reaches its ideal destination to *within one-half of one pixel spacing*,[1] and
- the actual address transformation (that is, including the roundoff) is *invertible*.[2]

The attendant image degradation is equivalent to a slight amount of diffusion (random walk with a mean free path on the order of one pixel spacing). This data-blind, token-reshuffling approach becomes particularly attractive with images having high spatial resolution but only few states per pixel—such as dithered images.

In spite of its minimal impact for one rotation step, diffusion would build up as rotations—each consisting of three shears—follow one another (for example, in animation, or when interactively inspecting an object from a variety of different angles). One obvious solution is to keep a copy of the entire original image; in this case, to rotate by R_2 after rotating by R_1 one would go back to the original and rotate it *in one step* by $R = R_2R_1$, paying the "diffusion fee" only once. The advantage of an invertible rotation algorithm is that it allows one to trade off processing for storage. Namely, one can memorize just the original angle parameters (rather than the entire image); after rotating by R_1, one rotates by R_1^{-1} to go back to the original and then applies $R = R_2R_1$ as before.

5 Shears and rotations in CAM-8

The rotation technique described here is ideally suited to CAM-8—a very fine-grained, parallel, uniform, scalable architecture offering unprecedented performance in the fine-grained modeling of spatially-extended systems[4]. In CAM-8 there are no "processors"; DRAM data are continually streamed through SRAMs, and the "program" regulates this traffic at the memory-controller level.

[1] Thus, after the first shear of (7) the error box will be, in pixel units, $(\pm 1/2, \pm 1/2, 0)$; after the second, $(\pm 1/2, \pm 1, \pm 1/2)$; and at the end of the rotation, $(\pm 1, \pm 3/2, \pm 1/2)$.

[2] To insure invertibility, compute the relative movement $\Delta\mathbf{r}$ of each pixel for each shear as a function of the Euler angles, using ordinary round-off to get integer components; to invert, perform the *same calculations* with the *same round-off rules*, and just perform the opposite movements $-\Delta\mathbf{r}$ in the inverse order.

A CAM-8 site is addressed by an n-dimensional integer vector (n can be larger than 3). Several distinct n-dimensional "layers" can be scanned in lockstep; at each site, data from all these layers can be combined and redistributed to the layers themselves; in particular, they can be copied from one layer to another with an arbitrary n-dimensional offset. In the following discussion we shall take $n = 3$ for definiteness.

Let $u = (u^1, u^2, u^3)$ be the *destination* site and $v = (v^1, v^2, v^3)$ the *source* site for a copy operation. The address offset (or "kick") k needed to bring source data from v to its destination u is $k = u - v$.

In the case we are considering here, the parameters of data movement from v to u are specified by an invertible linear transformation A that specifies a source site for each destination site, namely, $v = Au$, leading to a kick of the form

$$k = u - v = (I - A)v = Ku.$$

Thus, the kick specifications for the three shears are given by the matrices

$$K_i = I - S_i^{-1} \quad (i = 1, 2, 3), \tag{8}$$

each having just three nonzero entries. These entries are computed by the host computer (a conventional workstation) just once for a given rotation, and downloaded to the memory controllers. These, in turn, issue explicit memory addresses on-the-fly and appropriately route the corresponding data. The shears of Fig. 2 were produced in this way on CAM-8.[3]

6 Generalization to n dimensions and to $\mathrm{SL}(n)$ operators

The present three-shear decomposition generalizes to more than three dimensions and to more general matrices than rotations. Namely, every matrix of unit determinant (except for a subset of measure zero) can be written in the form $U_0 L U_1$, where the Us are upper triangular matrices and L a lower triangular—all three with unit diagonal. There is a simple canonical form for L that makes this decomposition unique. These results, which we present in detail elsewhere[10], are briefly summarized below.

6.1 General shears

The most general kind of linear shear in n dimensions is best visualized in a recursive way. In three dimensions, uniformly slice the space into two-dimensional "lasagna" sheets and slide these sheets along one another, using a constant offset from each sheet to the next; then slice each sheet into "spaghetti" strips and slide these strips along one another, using a constant offset from each strip to the next. In n dimensions, first slice the space into $(n-1)$-dimensional hyperplanes and shear the resulting structure; repeat the procedure going down one dimension at a time until the "spaghetti" stage is reached. Clearly, a shear is volume-conserving and thus has unit determinant.

If the slicing at the successive stages is done perpendicular to the coordinate axes $x_1, x_2, \ldots, x_{n-1}$ in that order, the overall operation is expressed by a *lower triangular matrix with unit diagonal*; if the axes are used in the reverse order, namely, $x_n, x_{n-1}, \ldots x_2$, then the matrix will be *upper* triangular. For brevity, in the rest of this paper the term **shear** will be restricted to such triangular matrices with unit diagonal.

6.2 Lemmas

Let $\delta_i(A)$ denote the determinant of the lower-left square submatrix of order i of matrix A. Two matrices A, B for which

$$\delta_i(B) = \delta_i(A) \qquad (i = 1, \ldots, n) \tag{9}$$

will be called **cognate**. A matrix A such that

$$\delta_i(A) \neq 0 \qquad (i = 1, \ldots, n) \tag{10}$$

will be called **well-born**.

Lemma 0.1 (Gauss) *If $B = UAV$, where A is square of order n and U and V are upper shears, then A and B are cognate.*

Lemma 0.2 (Gauss) *Given a well-born matrix A and a cognate matrix B, there exist unique upper shears U, V such that $A = UBV$.*

[3] In Fig. 2, object, shears, and illumination were simulated verbatim within a CAM-8 unit, at a resolution of 256×256×256 voxels (the "cubic frame" object is 128 voxels across). Voxels are binary (1="opaque solid", 0="vacuum"). The object was created as a smooth three-dimensional solid and then given a spongy 3-D texture by clearing a small fraction of the voxels at random.

The shears were pure address manipulation (no resampling was done after each shear). The shear amounts given by (8) were rounded to the nearest integer. In the machine, the axes x, y, z (z is "through the page" in the figure) are mapped into axes 1,3,2 of (7), so that the maximum error of 3/2 pixels at the end of the rotation (see Footnote 1) occurs along the z direction; this is the most visible striation seen in the fourth panel.

Rendering was done by illuminating the object, again cellular-automata-machine style, with a stream of discrete "photons" that were absorbed or scattered with a certain probability by the solid; the photons scattered in the direction of the viewer were accumulated. Illumination was from front-upper-left; since periodic boundary conditions were used in the x and y directions, in the second panel of the figure the lower-left corner of the object receives an artifact shadow from the upper-right corner of "itself" (i.e., its left-neighbor alias).

Lemma 0.3 (Kazhdan) *Every well-born matrix A of order n has a* unique *cognate of the following* **canonical** *form*

$$K = \begin{bmatrix} \kappa_n & & & & & & & & & & \\ \kappa_{n-1} & 1 & & & & & & & & & \\ \kappa_{n-2} & & 1 & & & & & & & & \\ \vdots & & & \ddots & & & & & & & \\ & & & & 1 & & & & & & \\ & & & & & 1 & & & & & \\ & & & & & 1 & 1 & & & & \\ & & & & 1 & & & 1 & & & \\ \vdots & & & \iddots & & & & & \ddots & & \\ \kappa_2 & & 1 & & & & & & & 1 & \\ \kappa_1 & 1 & & & & & & & & & 1 \end{bmatrix}, \quad (11)$$

where all elements are zero except the ones indicated (explicitly or by ellipses).

Lemma 0.4 *Given any nonsingular matrix A, there exists a permutation matrix P such that PA is well-born.*

6.3 Results

A **unit** matrix is a matrix having unit determinant, i.e., an element of SL(n); these include orthogonal matrices (rotations).

Theorem 0.5 *Any unit matrix A, if well-born, can be decomposed into three shears, $A = ULV$. There is a canonical such decomposition $A = UKV$, with K as in (11) with $\kappa_n = 1$.*

Remark 0.6 When the above canonical decomposition is specialized to 3-D rotations, it fails (because A is not well-born) on the two hyperplanes respectively given by

$$\gamma = \pi/2 \quad \text{and} \quad \alpha = \pi/2,$$

while the decompositon (6) of §3 fails on the hyperplanes

$$\alpha + \gamma = \pi \quad \text{and} \quad \beta = \pi.$$

Decomposition (6) yields simpler matrix elements as a function of the Euler angles α, β, and γ, and is thus somewhat more convenient for practical purposes.

Matrices that are not well-born form a subset of measure zero. In order to include also this exceptional subset, one can strike off the 'well-born' clause in Theorem 0.5, but then 'any unit matrix' must be qualified by 'up to a row permutation and possible sign adjustment'. Namely,

Theorem 0.7 *For any unit matrix A there exists a permutation matrix P such that either PA or $-PA$ can be decomposed into three shears, $A = ULV$.*

Another way to deal with the exceptional cases is to add a fourth shear.

Theorem 0.8 (Strang[7]) *Every unit matrix can be written as the product of four shears, $L_1U_1L_2U_2$.*

Finally, Strang's paper[7] (in press) describes a different three-shear decomposition of (almost all) unit matrices.

7 Conclusions

After recalling the advantages of factoring geometrical transformations of pixel arrays into shears, we have shown two ways to decompose a 3-D rotation into just three shears, namely, the algorithm of Fig. 3 and the canonical decomposition of Theorem 0.5. The latter construction generalizes to any number of dimensions and to more general transformations than rotations, namely, SL(n) transformations.

Acknowlegments

This research was supported in part by ARPA (ONR N00014-94-1-0662) and in part by NSF (9305227-DMS).

The first author (Toffoli) is very grateful to David Kazhdan, who substantially gave him the results of §6 in their entirety, and is indebted to Robert Kotiuga and Gilbert Strang for stimulus and advice. Norman Margolus provided the original stimulus and useful discussions. Suggestions by three anonymous referees led to a number of substantial improvements. Finally, we are greatly indebted to the journal's area editor Nelson Max for a superb re-reading that eliminated severe blemishes and led to many improvements.

References

[1] E. Catmull and A. R. Smith, 3-D transformations of images in scanline order, *ACM Computer Graphics* (SIGGRAPH) **14**, 1980, 279–285.

[2] Pat Hanrahan, Three-pass affine transforms for volume rendering, *Computer Graphics* **24**:5, 71–77, 1990.

[3] David Kazhdan, personal communication, June 1995.

[4] Norman Margolus and Tommaso Toffoli, Cellular Automata Machines, *Lattice Gas Methods for Partial Differential Equations* (Gary Doolen *et al.*, Ed.), pp. 219–248, Addison-Wesley, 1988.

[5] Alan Paeth, A fast algorithm for general raster rotation, *Proceedings, Graphics Interface '86*, pp 77–81, Canadian Information Processing Society, Vancouver, 1986. Reprinted with corrections and additions in *Graphics Gems* (Andrew Glassner, Ed.), pp 179–195, Academic Press, New York, 1990.

[6] A. R. Smith, Planar 2-pass texture mapping and warping, *ACM Computer Graphics* (SIGGRAPH) **21**, 263–272, (1987).

[7] Gilbert Strang, Every unit matrix is a *LULU*, to appear in *Linear Algebra and Its Applications*.

[8] A. Tanaka, M. Kameyama, S. Kazama, and O. Watanabe, A rotation method for raster image using skew transformations, *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp 272–277, IEEE, 1986.

[9] Philippe Thévenaz and Michael Unser, Efficient geometric transformations and 3-D image registration, *ICASSP'95*, pp 2919–2922, IEEE, 1995.

[10] Tommaso Toffoli, Almost every unit matrix is a ULU, *Linear Algebra and Its Applications* **259** (1997), 31–38.

[11] Klaus Voss, *Discrete Images, Objects, and Functions in* Z^N, Springer-Verlag, 1993.

[12] George Wolberg, *Digital Image Warping*, IEEE Computer Society Press, Los Alamitos, CA, 1990.